# MULTIPLE OBJECT TRACKING WITH OCCLUSIONS USING HOG DESCRIPTORS AND MULTI RESOLUTION IMAGES

**Piotr BILINSKI, Francois BREMOND, Mohamed Becha KAANICHE**

INRIA Sophia Antipolis – Mediterranean Research Center – PULSAR team
2004 Route des Lucioles, BP 93, 06902 Sophia Antipolis, France
E-mail: {Piotr.Bilinski, Francois.Bremond, Mohammed-Becha.Kaaniche}@sophia.inria.fr

**Keywords:** Multiple object tracking, occlusions, Histogram of Oriented Gradients, multi resolution images, Features from Accelerated Segment Test.

## Abstract

*We present an algorithm for tracking multiple objects through occlusions. Firstly, for each detected object we compute feature points using the FAST algorithm [1]. Secondly, for each feature point we build a descriptor based on the Histogram of Oriented Gradients (HOG) [2]. Thirdly, we track feature points using these descriptors. Object tracking is possible even if objects are occluded. If few objects are merged and detected as a single one, we assign each newly detected feature point in such single object to one of these occluded objects. We apply probabilistic methods for this task, using information from the previous frames like object size and motion (speed and orientation). We use multi resolution images to decrease the processing time. Our approach is tested on the synthetic video sequence, the KTH dataset [3] and the CAVIAR dataset [4]. All tests confirm the effectiveness of our approach.*

## 1 Introduction

Most of the computer vision applications like surveillance systems use tracking algorithms. Despite the fact that many different approaches have been proposed in the last decades, multiple object tracking through occlusion is still one of the most challenging issues in the computer vision. There are a lot of difficulties for a single object tracking like illumination variability, background noise and occlusions. Multiple object tracking is even more challenging due to multi object occlusions.

In the real time applications the tracking algorithm can use only a small part of the processing time assigned to one frame of the video sequence. Most of the time is assigned to image processing stages like segmentation or high-level stages such as action recognition. For this reason, algorithms tracking objects have to be very efficient. Taking these issues into consideration, we propose an efficient approach tracking multiple objects and handling occlusions. Our algorithm computes for each detected object feature points (corner points) using the Features from Accelerated Segment Test (FAST) corner detector algorithm [1]. Once, feature points are found, we track them in the next frames comparing their HOG descriptors values. By computing mean HOG descriptors the tracking process is much more robust under occlusions. To speed up the processing time for searching the most similar point, we use multi resolution images. Thanks to this solution we are able to track multiple objects in the real time.

Sometimes a group of objects is detected as a single object. The reason can be segmentation process or object occlusions. We consider such situations by approximating positions of partially occluded objects and by assigning each newly found feature point in this single object to the one of these occluded objects. To do this, we apply probabilistic methods using the information like object size and object motion (speed and orientation).

Our approach is tested on three video sequences. The first one is the synthetic sequence, the second one is the KTH dataset [3] and the third one is the CAVIAR dataset [4]. All tests confirm the effectiveness of our approach.

The paper is organized as follows. In section 2, we present an overview of previous research on object tracking. In section 3, we present in details our multiple object tracker. Section 4 describes the method of handling with occlusions. We present how we track occluded objects and how we assign each newly detected feature point to the occluded object. Section 6 describes a few experiments and their results. In section 7, we summarize the contributions of this paper and show potential future research directions.

## 2 Previous work

For the past two decades, many approaches have been proposed for multiple object tracking handling occlusions. Rehg and Kanade [5] propose an object tracker which uses a kinematic model to predict occlusions. They use windowed templates to track partially occluded objects. Isard and MacCormick [6] present the Bayesian multiple-blob tracker handling occlusions. Khan and Shah [7] propose a tracker using the Expec-

tation Maximization algorithm and the maximum a posteriori probability approach. Cavallaro et al. [8] propose an object tracker based on a hybrid strategy using both object and region information. In their approach low-level features are extracted. Cucchiara et al. [9] propose an object tracker which exploits a probabilistic function and an appearance model. Bak et al. [10] propose a method to fuse information from the motion segmentation with online adaptive neural classifier for a robust object tracking. Kaaniche and Bremond [11] propose the HOG tracker for Gesture Recognition using the Kalman filter to predict a new position of feature points. We built our own tracker based on this last approach. We improved the quality of the tracker changing the method of building, updating and matching the descriptors. We extended the tracker to work with multiple objects handling occlusions and we reduced the processing time of the tracker using multi resolution images.

# 3 Moving object tracking

The goal of this section is to present an algorithm tracking multiple object and working with occlusions. For each video frame our algorithm works in the following steps:

1. Moving object detection (section 3.1).
2. New feature point detection (section 3.2) for each detected object.
3. 2D HOG descriptor building (section 3.3) for each newly detected feature point.
4. 2D HOG descriptor tracking (section 3.4) in the next frames of the video sequence.

## 3.1 Moving object detection

To detect objects in the current frame we compute the difference between the current image and the reference (background) image [12]. Then, we form moving regions by grouping foreground neighbor pixels. Finally, the moving regions are classified into objects by classifiers based on the object size.

## 3.2 New feature point detection

Once, the objects are detected in the current scene, we apply for each of them FAST algorithm [1]. Corner detector is used to get for each object a set of feature points. Instead of FAST [1] any other corner detector algorithm like Shi-Tomasi [13] can be applied. Our method sort detected points in a descending order using corner strength information. Then, starting from the most significant point (one with the biggest value of the corner strength), we choose a subset of points which firstly ensures a minimum distance between feature points in this subset and secondly ensures a minimum distance between feature points in this subset and all tracked points in this object. The minimum distance between points improves points distribution in an object and helps to prevent overlapping of tracked points.

## 3.3 2D HOG descriptor building

Once, the feature points are computed, we build for each of them a descriptor based on Histogram of Oriented Gradients (HOG) [2]. To compute such a descriptor we convert color images to gray scale intensity images. Then, for each feature point we define a block which is a square containing 9 ($3 \times 3$) cells. Each cell is a square containing $n \times n$ (where $n$ is equal to 3 or 5) pixels. The center of this block is situated on the position of the feature point. For each pixel in this block we apply the Sobel operator [14] to compute the approximate absolute gradient magnitude (normalized to values between 0 and 1) and gradient orientation. Then, for each cell we define $K (K = 9)$ orientation bins and using the gradient orientation, we assign each pixel in a cell to one bin. For each bin we calculate a sum of gradients magnitude of its pixel. It means that for each cell inside a block we obtain a vector of 9 values (sums). The 2D HOG descriptor is a vector $d$ (normalized to values between 0 and 1) concatenating vectors of all cells in the block.

## 3.4 2D HOG descriptor tracking

Once the feature point $A$ is computed we track it in the next frames doing the following steps:

1. Computing the searching radius (section 3.4.1).
2. Tracking in the low resolution image (section 3.4.2).
3. Tracking in the high resolution image (section 3.4.3).
4. Updating the descriptor (section 3.4.4).

### 3.4.1 Computing the searching radius

We compute the geometric mean $S_{mean}$ of object speed taking into account all the speed values of points (belonging to the same object as the point $A$) from the previous frames. If the point is newly detected, we assume that $R_{mean}$ is equal to the maximal speed $R_{MAX}$ (parameter of the algorithm) of the object. We approximate the searching radius $R_H$ which is defined as:

$$R_H = R_{mean} + \frac{1}{T} \times (R_{MAX} - R_{mean}) \qquad (1)$$

where $T$ is the number of frames for which the point $A$ was tracked. Thanks to this equation the longer we track the feature point, the more accurate we are in approximating the searching radius.

If the point is not found in the current frame, we will repeat the tracking part assuming that $R_H$ is equal to $R_{MAX}$. If we find then searching point, we will assume in the next frame (only for computing the searching radius) that there was no tracking till this frame.

To optimize the computation time of the geometric mean we represent it as a sum of logarithmic values. Assuming that $n$ is the number of frames in which the point $A$ was tracked, we can compute the geometric mean value in a new frame in time $\Theta(1)$ instead of $\Theta(n)$.

#### 3.4.2 Tracking in the low resolution image

We convert high resolution image to the low resolution image. Then, we consider in the low resolution image all such points which belong to an object and they are they not further from the point $A$ than the distance $R_L$ (which is in the low resolution image an equivalent value of a distance $R_H$). Then we compute the differences between all these points and the point $A$ and we choose the one with the smallest value. If this value is bigger than the threshold $T_{DIFF-BW-DESC}$ (parameter of the algorithm), we assume that we lost the point. Otherwise, we assume that the point $A$ is tracked in the low resolution image as a point $P_L$.

To compute the difference between the point $A$ and the point $B$ we compare their HOG descriptors ($d^A$ and $d^B$). Such difference between the two HOG descriptors is defined by the function $E(d^A, d^B)$ given by the equation:

$$E(d^A, d^B) = \sum_{i=1}^{9\times K} [v_i^B \times (d_i^A - d_i^B)^2] \quad (2)$$

where $v$ is a vector of variability computed for each element of the corresponding descriptor. For each newly detected feature point elements of the vector $v$ are equal to 1.

#### 3.4.3 Tracking in the high resolution image

Once, we computed that point $A$ is tracked in the low resolution image as point $P_L$, we consider all corresponding (to the point $P_L$) points in the high resolution image which belong to an object and are not further from the point $A$ than the distance $R_H$ (parameter $R_H$ is defined by the equation 1). Like in the low resolution image we compute the differences between all these points and the point $A$ and we choose one with the smallest value. If this value is bigger than the threshold $T_{DIFF-BW-DESC}$ or it is bigger than value of the next similar point in the low resolution image, we consider the next similar point in the low resolution image and we start tracking in the high resolution image part from the beginning (if there is no more points we assume that we lost the point $A$). Otherwise we assume that the point $A$ is tracked in the high resolution image as a point $P_H$. To compute the difference between the point $A$ and the point $B$ we use the equation 2.

#### 3.4.4 Updating the descriptor

We assume in this section that in the high resolution image point $A$ from the previous frame is tracked as a point $B$ in the current frame. We compute the mean HOG descriptor for the point $B$ ($\hat{d_i^B}$) updating the mean HOG descriptor of the point $A$ ($\hat{d_i^A}$) by the corresponding HOG descriptor of the point $B$ ($d_i^B$):

$$\forall_{i=1..9\times K} \quad \hat{d_i^B} = \alpha \times \hat{d_i^A} + (1-\alpha) \times d_i^B \quad (3)$$

where $\alpha$ is a parameter describing how big influence have the descriptor $d^B$ on the mean descriptor $\hat{d^B}$ computed for the same point.

After updating the descriptor we also compute the variability vector $\hat{v^B}$ for each element of the mean HOG descriptor $\hat{d^B}$:

$$\forall_{i=1..9\times K} \quad \hat{v_i^B} = \alpha \times \hat{v_i^A} + (1-\alpha) \times |\hat{d_i^A} - d_i^B| \quad (4)$$

## 4 Object occlusions

If $k$ separated objects are merging into one larger object we split such single object into the $k$ separated (but occluded) objects and we assign newly detected feature points in it into one of these $k$ objects. For splitting we assume that:

- $A$ – is the set of detected objects in the previous frame.
- $B$ – is the set of detected objects in the current frame.
- $n_a$ – is the number of elements in the set $A$.
- $n_b$ – is the number of elements in the set $B$.
- $A_i$ – is the object from the set $A$ ($1 \leq i \leq n_a$).
- $B_j$ – is the object from the set $B$ ($1 \leq j \leq n_b$).
- $O$ – is the set of objects which were computed in the previous frame as separated objects and are computed in the current object as a single object.
- $k$ – is the number of elements in the set $O$.
- $O_i$, $O_z$ – are objects from the set $O$ ($1 \leq i, z, \leq k$).

### 4.1 Merging objects detection

In order not to take into consideration single, badly tracked points, we assume that the object is tracked if the number of its tracked points is bigger than threshold $T_{MIN-P-NB}$ (parameter of the algorithm). We define such function $Y(A_i, B_j)$ which returns 0 if the number of the tracked points is less than the threshold $T_{MIN-P-NB}$ and 1 otherwise. Thanks to this function we can compute how many objects were detected in the previous frame as separated objects and are detected in the current frame as a single object $B_j$:

$$N(B_j) = \sum_{i=1}^{n_a} Y(A_i, B_j) \quad (5)$$

### 4.2 Algorithm assigning newly detected point to the merged object

We assume that we have detected that $k$ separated in the previous frame objects are detected in the current frame as a single object. We assign then each newly detected in this single object feature point $P$ into one of these $k$ objects ($O_i$). In this order we do the following steps:

- In the current frame $f$: we compute the probability for assigning point $P$ to the object $O_i$ using a bounding box method described in section 4.2.1. We choose one occluded object for which the probability value is the biggest. It is an initial assignment.

- In the next frame $f+1$, we compute probabilities for assigning point $P$ to the object $O_i$ using three methods (a bounding box, a speed and an orientation method) which are described in sections: 4.2.1, 4.2.2 and 4.2.3. For each occluded object, we compute the geometric mean of all these probabilities that we can compute, taking into account also the probability computed in the frame $f$. We choose the object with the biggest mean probability. Since now, the feature point is assigned to the object.

The method proposed in section 4.2.1 can also be used to approximate the sides of the bounding box of the tracked occluded object.

#### 4.2.1 Bounding Box

To compute the probability that point $P$ belongs in the current frame to the object $O_i$ we assume that:

- $B_{left}(R, O_i)$ – is a function which returns geometric mean of distance between point $R$ (belonging to the object $O_i$) and the left side of the bounding box of the object $O_i$ computed through all frames where the point $R$ was tracked.
- $B_{left}(O_i)$ – is a function which approximates the left side of the bounding box of the object $O_i$ computing the geometric mean of values:

$$R_x - B_{left}(R, O_i) \tag{6}$$

  through all tracked in the current frame points $R(R_x, R_y)$ which belong to the object $O_i$.

In the same way we define functions: $B_{right}(R, O_i)$, $B_{top}(R, O_i)$ and $B_{bottom}(R, O_i)$ which allows to approximate the rest sides ($B_{right}(O_i)$, $B_{top}(O_i)$, $B_{bottom}(O_i)$) of the bounding box of the object $O_i$.
Thanks to these definitions we can calculate the distance between the point $P(P_x, P_y)$ and the bounding box of the object $O_i$ using formula:

$$D_B(P, O_i) = \sqrt{D_x(P_x, O_i)^2 + D_y(P_y, O_i)^2} + e \tag{7}$$

where:

- $D_x(x, O_i)$ equals 0 if $B_{left}(O_i) \leq x \leq B_{right}(O_i)$ and equal to:

$$min(|B_{left}(O_i) - x|, |B_{right}(O_i) - x|) \tag{8}$$

  otherwise.
- $D_y(y, O_i)$ equals 0 if $B_{top}(O_i) \leq y \leq B_{bottom}(O_i)$ and equal to:

$$min(|B_{bottom}(O_i) - y|, |B_{top}(O_i) - y|) \tag{9}$$

  otherwise.
- $e$ is a very small number (i.e. $1^{-100}$) used to avoid the division by zero situations.

The probability that the point $P$ belongs to the bounding box of the object $O_i$ is defined by following formula:

$$P_B(P, O_i) = 1 - \frac{D_B(P, O_i)}{\sum_{z=1}^{k} D_B(P, O_z)} \tag{10}$$

#### 4.2.2 Speed

To compute the probability that the point $P$ belongs to the object $O_i$ in the current frame we assume that:

- $SP_{mean}(R)$ – is for the point $R$ the geometric mean of speed computed for all the previous frames.
- $SO_{mean}(O_i)$ – is the geometric mean of speed computed for values $SP_{mean}(R)$ of all tracked in the current frame points $R$ which belongs to the object $O_i$.
- $D_S(R, O_i)$ – is the distance function between the point $R$ and the object $O_i$ defined by the formula:

$$D_S(R, O_i) = |SP_{mean}(R) - SO_{mean}(O_i)| + e \tag{11}$$

  where $e$ is a very small number (i.e. $1^{-100}$) used to avoid the division by zero situations.

Thanks to these assumptions we can define the probability that the point $P$ belongs to the object $O_i$:

$$P_S(P, O_i) = 1 - \frac{D_S(P, O_i)}{\sum_{z=1}^{k} D_S(P, O_z)} \tag{12}$$

#### 4.2.3 Orientation

The probability that the point $P$ belongs to the object $O_i$ is defined in the similar way as in the section 4.2.2 but instead of speed we consider the orientation parameter.

## 5 Experiments and results

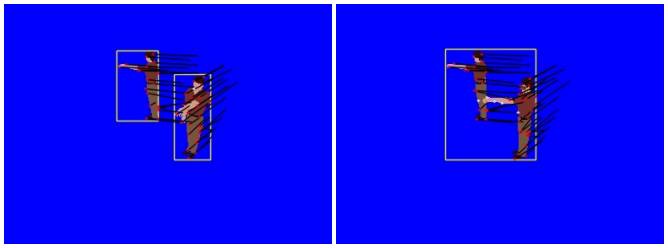
We have tested our approach on three video sequences: a synthetic sequence, a sequence from the KTH dataset [3] and a sequence from the CAVIAR dataset [4]. In all figures presented in this section:

- The yellow rectangles correspond to the bounding boxes of the detected objects.
- The red points presents tracked feature points.
- The white points correspond to the newly detected feature points.
- The black lines are the trajectories of feature points.

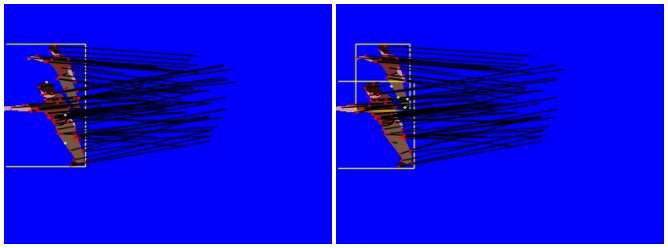
In all experiments we assumed that we consider such objects for which all sides of the bounding box are bigger than 10. We also assumed that a minimum distance between corner points is 9, $R_{MAX}$ parameter is equal to 9, $T_{MIN-P-NB}$ is equal to 2, $\alpha$ is equal to 0.6 and $T_{DIFF-BW-DESC}$ parameter is equal to $\frac{9 \times K}{100} = 0.81$. These criteria were chosen experimentally.

### 5.1 Synthetic sequence

The generated video sequence presents two objects which are at the beginning of the sequence detected as separated, then as a single object and after some frames again as separated objects. The results of this experiment are presented on the figure 1. The figure shows that most of the points are tracked correctly that let us to track objects through partial occlusions with success.



(a) Frame: 12. (b) Frame: 13.



(c) Frame: 40. (d) Frame: 41.

Figure 1: This figure shows how our algorithm works before (1a) and after (1b) merging and before (1c) and after (1d) splitting merged objects.

### 5.2 KTH database

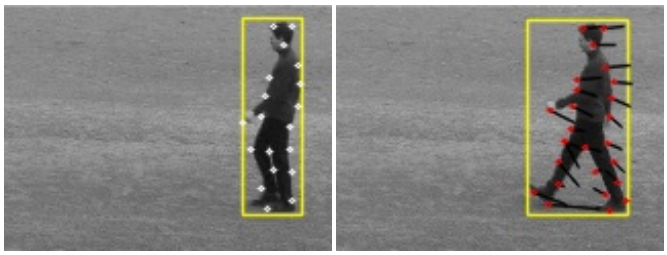
The KTH [3] video database contains 600 video files with spatial resolution of $160 \times 120$ pixels. The database contains six types of human actions (walking, jogging, running, boxing, hand waving and hand clapping) which were recorded several times by 25 subjects in four different scenarios (outdoors, outdoors with scale variation, outdoors with different clothes and indoors). They were recorded with the homogeneous background and a static camera with 25 fps.

In the second experiment we show the effectiveness of our algorithm using multi resolution images. In our test case scenario we chose ratio 1:2 between the low and the high resolution images. It means that 4 ($2 \times 2$) pixels in the high resolution image are converted to one pixel in the low resolution image. In our experiment we compared two algorithms. The first one (Algorithm 1) uses the high resolution images and checks all points in a distance not further than $R_{MAX}$. The second one (Algorithm 2) is described in section 3.4. For the experiment we assumed that $R_H$ is always equal to $R_{MAX}$. The results presented in the table 1 show that using multi resolution images the number of compared HOG descriptors is significantly decreased which also means that the processing time is decreased.

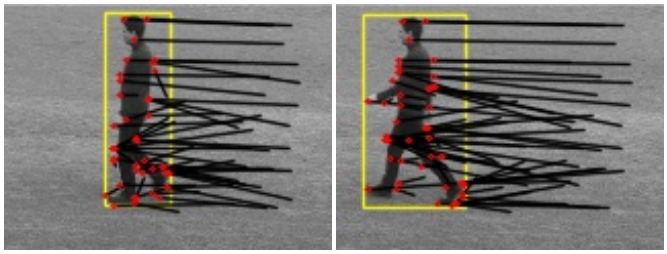
| | Minimal | Maximal | Mean |
|---|---|---|---|
| Algorithm 1 | 4 | 196 | 110.71 |
| Algorithm 2 | 5 | 68 | 40.86 |
| Algorithm 2 / Algorithm 1 | 1.25 | 0.35 | 0.37 |

Table 1: Minimal, maximal and mean (arithmetic) number of compared HOG descriptors computed for all detected feature points in the video sequence. In the last row of this table, we divided the values obtained for the algorithm 2 over the values obtained for the algorithm 1. The second algorithm is slower only in a few situations, but the difference is not big and such situations exist rarely.

During this experiment we obtained the following statistics: arithmetic mean of number of tracked points per frame: 37.26, arithmetic mean of number of lost points per frame: 0.1, arithmetic mean of number of all (newly detected and tracked) points per frame: 38.48. Statistics indicate that our algorithm performs better than Kanniche and Bremond [11]. We also have to consider the output from our approach (figure 2), where some points are moving so radical that we can say that they are badly tracked. However, most of the points are tracked correctly that let us to track objects with success.



(a) Frame: 1. (b) Frame: 5.



(c) Frame: 27. (d) Frame: 37.

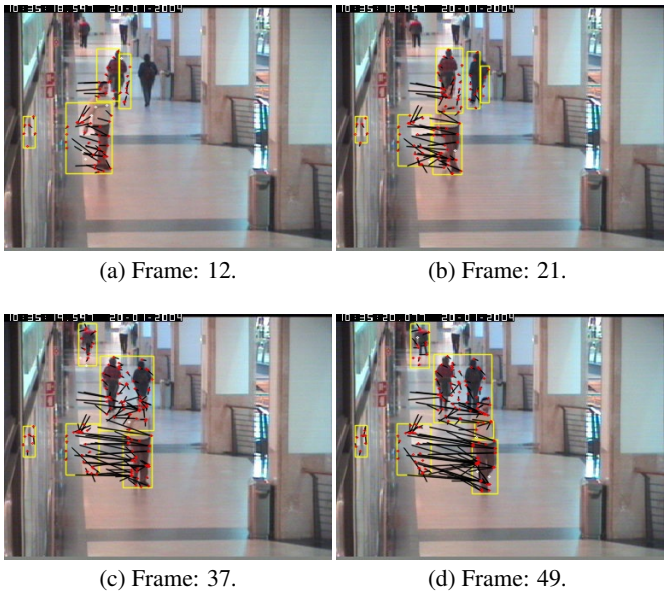
Figure 2: KTH dataset. The output of our tracker.

### 5.3 CAVIAR database

The CAVIAR [4] video database contains two groups of video sequences. The first group of video clips was recorded at the entrance lobby of the INRIA Labs in Grenoble (France). The second one was recorded along and across the hallway (two synchronized videos for each sequence) in a shopping center in Lisbon (Portugal). This database contains different scenar-

ios such as people walking alone, meeting the others, window shopping, entering and exiting shops, fighting, passing out or leaving a package in a public place. All video sequences were recorded with a wide angle camera lens in half-resolution PAL standard (384 $\times$ 288 pixels, 25 fps) and compressed using MPEG2.

In the third experiment we chose from this database a short video sequence with a partial occlusion. This video sequence presents one person leaving the shop and two others going through the hall. The results are presented on the following figure 3. The figure shows that some points are lost, some points are tracked incorrectly but most of the points are tracked correctly that let us to track objects through partial occlusions with success.



(a) Frame: 12. (b) Frame: 21.

(c) Frame: 37. (d) Frame: 49.

Figure 3: CAVIAR dataset. The output of our tracker.

## 6 Conclusions

We have presented a novel and a fast multiple object tracker handling occlusions. The proposed tracker uses the FAST algorithm [1] to detect corner points, HOG descriptors to track feature points and multi resolution images to reduce the processing time. The results from all three experiments show the effectiveness of our approach. Our experiments confirm that the processing time is significantly decreased using multi resolution images. Our algorithm manages to track objects despite of occlusions.

For future research, we are going to improve the quality of the tracker using appearance features such as color, shape, size, etc. We would like to test our tracker to work with full occlusions.

## References


[1] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, volume 1, pages 430–443, 2006.

[2] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE International Conference on Computer Vision and Pattern Recognition*, volume 2, pages 886–893, 2005.

[3] KTH dataset. Website. http://www.nada.kth.se/cvap/actions/.

[4] CAVIAR dataset. Website. EC Funded CAVIAR project/IST 2001 37540. http://homepages.inf.ed.ac.uk/rbf/CAVIAR/.

[5] J. M. Rehg and T. Kanade. Model-based tracking of self-occluding articulated objects. *IEEE International Conference on Computer Vision*, 0:612–617, 1995.

[6] M. Isard and J. Maccormick. Bramble: a bayesian multiple-blob tracker. In *IEEE International Conference on Computer Vision*, volume 2, pages 34–41, 2001.

[7] S. Khan and M. Shah. Tracking people in presence of occlusion. In *Asian Conference on Computer Vision*, pages 1132–1137, 2000.

[8] A. Cavallaro, O. Steiger, and T. Ebrahimi. Tracking video objects in cluttered background. *IEEE Transactions on Circuits and Systems for Video Technology*, 15:575–584, 2005.

[9] R. Cucchiara, C. Grana, G. Tardini, and R. Vezzani. Probabilistic people tracking for occlusion handling. In *International Conference on Pattern Recognition*, pages 132–135, 2004.

[10] S. Bak, S. Suresh, F. Bremond, and M. Thonnat. Fusion of motion segmentation with online adaptive neural classifier for robust tracking. In *International Conference on Computer Vision Theory and Applications*, pages 410–416, 2009.

[11] M. B. Kaaniche and F. Bremond. Tracking hog descriptors for gesture recognition. In *IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2009.

[12] A. Nghiem, F. Bremond, and M. Thonnat. Shadow removal in indoor scenes. In *IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2008.

[13] J. Shi and C. Tomasi. Good features to track. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[14] A. Hornberg. Handbook of machine vision. *Wiley–VCH*, page 598, 2006.